Návrh

Exponát do Aurelia
Detekcia žmurkacích gest

Ing. Peter Varga

Miroslav Baluch, Matej Dráb, Ivana Nemsilajová, Martina Veselá

Obsah

1 Úvod

1.1 Účel dokumentu

Tento dokument slúži ako návrh informačného systému exponátu do Zážitkového centra vedy Aurelium. Dokument dôkladne popisuje funkcie a metódy informačného systému pomocou diagramov a popisov, spôsob ako bude systém vyvinutý a ako bude následne fungovať tak, aby spĺňal všetky požiadavky obsiahnuté v Katalógu požiadaviek.

2 Dekompozícia a popis

2.1 Python

Python je interpretačný programovací jazyk. Využíva objektovo-orientovaný prístup. Kód aplikácie bude napísaný v tomto jazyku. Budeme využívať najnovšiu verziu tohto programovacieho jazyka (3.7).

2.1.1 OpenCV

OpenCV je knižnica pre python na spracovanie a zaznamenávanie obrazu. Zároveň daná knižnica vie komunikovať s vonkajšími snímacími zariadeniami (v našom prípade ide o webkameru). Po nasnímaní obrazu sú jednotlivé snímky prevedené z RGB kódovania na čiernobiele kódovanie. Okrem toho nám táto knižnica dovoľuje pracovať s oknom aplikácie a umožňuje kreslenie jednoduchých tvarov a obrázkov, ktoré využijeme na označenie detekovanej časti.

2.1.1.1 OpenCV Funkcie

- `cv2.VideoCapture(int:numberOfCamera(starting with 0))` - sníma video z príslušnej kamery
- `cv2.putText(frame:frame,string:text)` - vypíše text
- `cv2.rectangle(frame:frame,tuple:coords,tuple:coords,color:color,int:width)` - nakreslí obdĺžnik na dané miesto
- `cv2.imshow(string:name,frame:frame)` - vykreslí daný obrázok
- `cv2.waitKey(int:number)==int:number` - čaká pokiaľ nie je zmačknutý kláves
- `cv2.destroyAllWindows()` - zahodí vytvorené okná
- `cv2.resize(frame:frame,tuple:size)` - mení veľkosť obrázka
- `cv2.cvtColor(frame:frame,color:color)` - vyberá farebné kódovanie (`cv2.COLOR_BGR2GRAY`)
- `cv2.line(frame:frame,tuple:tuple,tuple:tuple,color:color)` - nakreslí čiaru
- `cv2.bilateralFilter(frame:frame,int:distance)` - aplikuje bilateral filter
- `cv2.erode(frame:frame, algorithm:algo, int:iterations)` - eroduje obraz použitím štrukturálneho elementu
- `cv2.threshold(frame:frame, int:thresholdWhite, int:thresholdBlack, cv2.THRESH_BINARY)` - zmení farby obrazu na čiernu a bielu podľa obmedzenia
- `cv2.countNonZero(frame:frame)` - počet nonZero framov
- `cv2.fillPoly(frame:frame,tuple:points,color:color)` - vyplní polygón danou farbou
- `cv2.bitwise_not(frame:frame, frame:frame, mask:mask)` - prevráti všetky bity v zozname
- `cv2.moments(mask:mask)` - pomáha rozpoznať určité charakteristiky obrazu
- `cv2.findContours(frame:frame, cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)` - nájde všetky kontúry objektu
- `cv2.namedWindow(frame:frame,property:prop )` - vytvorí okno s danou vlastnosťou (`cv2.WND_PROP_FULLSCREEN`)
- `cv2.setWindowProperty(frame:frame, property:prop, property:prop, ...)` - pridá vlastnosť existujúcemu oknu
- `camera.read()` - vráti nový frame kamery
- `camera.get(property:prop)` - ukáže fps alebo iné vlastnosti
- `camera.isOpened()` - zistí, či je kamera aktívna

2.1.2 NumPy

Knižnica pre Python určená na prácu s viacrozmernými poľami a maticami.

2.1.2.1 NumPy Funkcie

- `numpy.ndarray(shape:shape,int:offset)` - multidimenzionálny zoznam prvkov fixnej veľkosti
- `numpy.ones(shape:shape,int:order)` - vytvorí zoznam daného tvaru a typu, vyplnený jednotkami
- `numpy.zeros((int:height, int:width), dtype:numpy.uint8)` -vytvorí zoznam daného tvaru a typu, vyplnený nulami
- `numpy.array(object:obj, dtype:type)` - vytvorí zoznam
- `numpy.max(region:fromto)` - maximum zoznamu elementov
- `numpy.min(region:fromto)` - minimum zoznamu elementov
- `np.full((int:height, int:width), int:value,dtype:type)`- vytvorí zoznam daného tvaru a typu, vyplnený danou hodnotou

2.1.3 GazeTracking

Knižnica pre Python, ktorá obsahuje naučený algoritmus pre snímanie tváre, očí a pohľadu doľava/doprava.

Modul containing pre-trained eye detection algorithm

2.2 CMake

Open-source software použitý na vybudovanie projektu z C++ do Python. Pre správne fungovanie sa vyžaduje C++ kompilátor vo vhodnej verzií (32/64 bit). U nášho projektu to slúži na nainštalovanie C++ modulov do Pythonu.

3 Návrh používateľského rozhrania

3.1 UI

Aplikácia bude zaberať celú obrazovku. Pokiaľ sa nezhoduje aspect ratio kamery a obrazovky, tak sa obrazovka doplní čiernymi barmi. V celom okne bude zobrazený živý obraz kamery.

Po detekovaní vhodnej tváre aplikácia túto tvár orámuje jemným rámikom. Týmto dá vedieť používateľovi, že ho kamera správne detekuje.

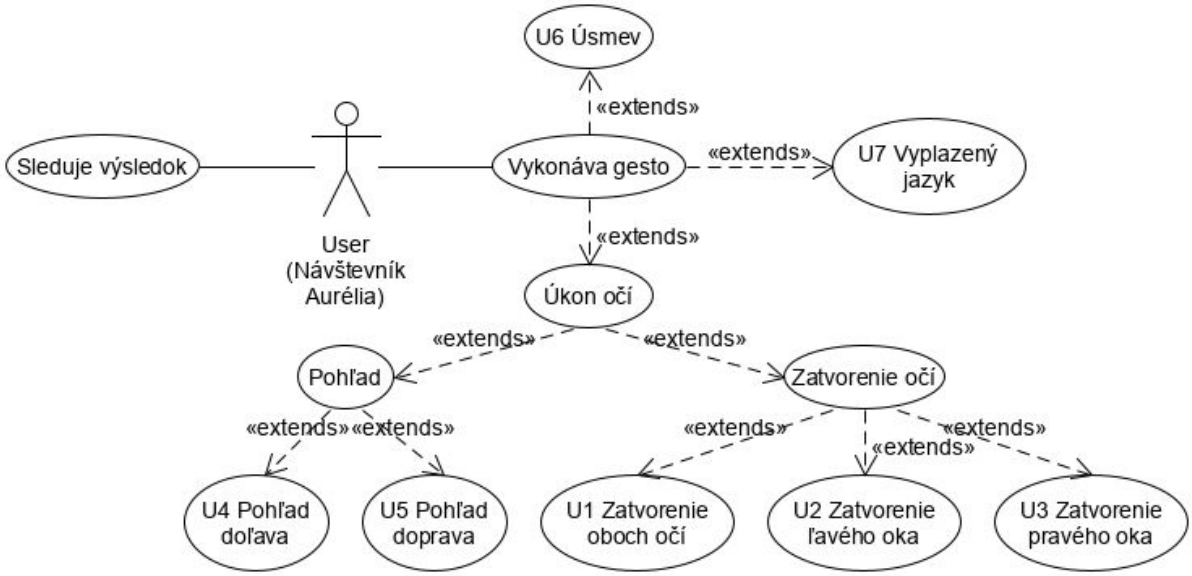
V pravom hornom rohu bude zobrazený momentálny úkon, ktorý používateľ vykonáva.

V ľavom hornom rohu sa na pár sekúnd po úspešnom vykonaní gesta zobrazí textový výpis nastaviteľný v konfiguračnom súbore.

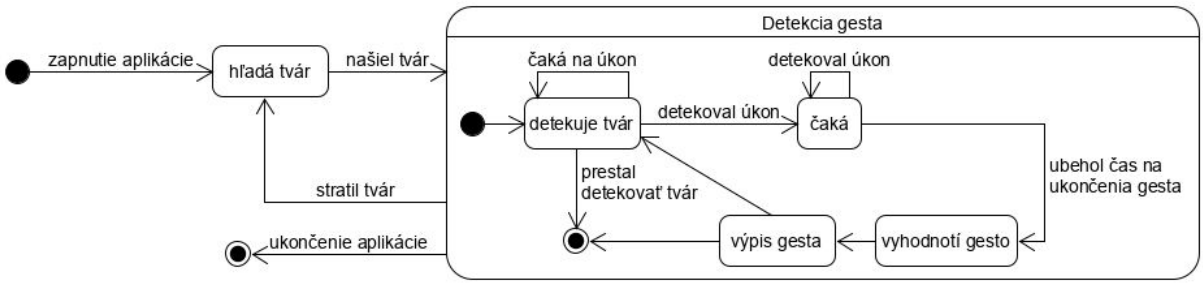
Na spodku aplikácie bude nápoveda gest. Bude prekrývať záznam kamery, jej plnej šírky, polopriehladným obdĺžnikom. Gestá sa na nej budú zobrazovať z ľava do prava. Pre každé nakonfigurované gesto bude zobrazené jeho meno a obrázky znázorňujúce postupnosť úkonov potrebnú na správne vykonanie gesta.

4 Návrh implementácie

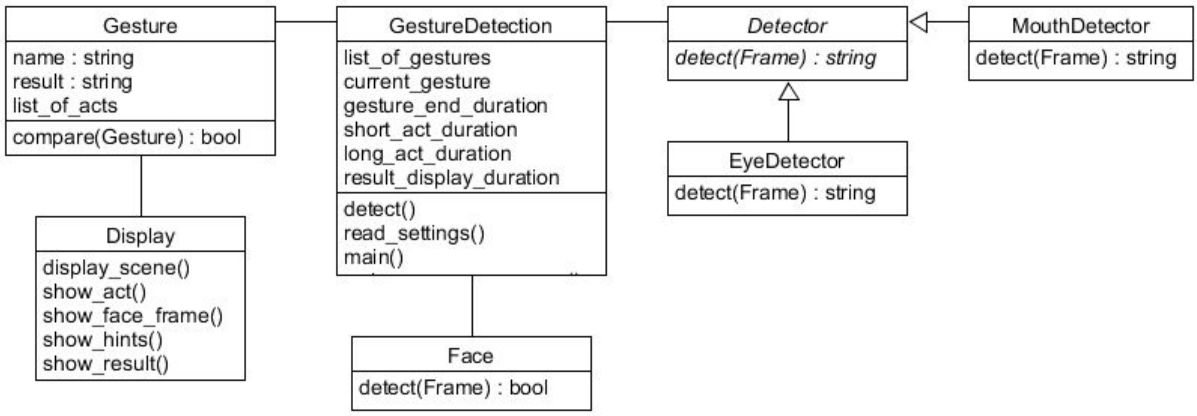
4.1 UML - Use-case diagram



4.2 UML - Stavový diagram



4.3 UML - Triedny diagram



`Gesture` - gesto

- `name` - meno daného gesta
- `result` - text priradený ku gestu, ktorý bude zobrazený po jeho správnom vykonaní
- `list_of_acts` - postupnosť úkonov potrebných na vykonanie gesta
- `compare(Gesture)` - porovná dve gestá a zistí, či sa zhodujú

`Display` - trieda obsahujúca funkcie, ktoré zobrazia elementy na obrazovke

- `display_scene()` - na celej obrazovke bude bežať živý obraz z kamery
- `show_act()` - v pravom hornom rohu zobrazí momentálne vykonávaný úkon
- `show_face_frame()` - tenkým rámikom orámuje detekovanú tvár
- `show_hints()` - na spodku obrazovky zobrazí nápovedu ako vykonať prednastavené gestá
- `show_result()` - v ľavom hornom rohu vypíše Gesture result

`GestureDetection`

- `list_of_gestures` - zoznam všetkých prednastavených gest, ktoré aplikácia rozpoznáva
- `current_gesture` - postupnosť momentálne vykonaných úkonov
- `gesture_end_duration` - čas čakania po poslednom úkone na ukončenie gesta
- `short_act_duration` - čas trvania krátkeho úkonu
- `long_act_duration` - čas trvania dlhého úkonu
- `result_display_duration` - dĺžka zobrazenia výpisu po vykonaní gesta
- `detect()` - zistí, či sa zadané gesto nachádza v zozname rozpoznateľných gést
- `read_settings()` - prečíta a nastaví nastavenia z konfiguračného súboru
- `main()` - tu bude bežať celý program
- `wake_up_screen_saver()` - vypne screen saver

`Face` - trieda na detekciu tváre

`detect(Frame)` - vráti, či momentálne detekuje tvár

`Detector` - abstraktná trieda

`detect(Frame)` - vráti meno rozpoznaného úkonu

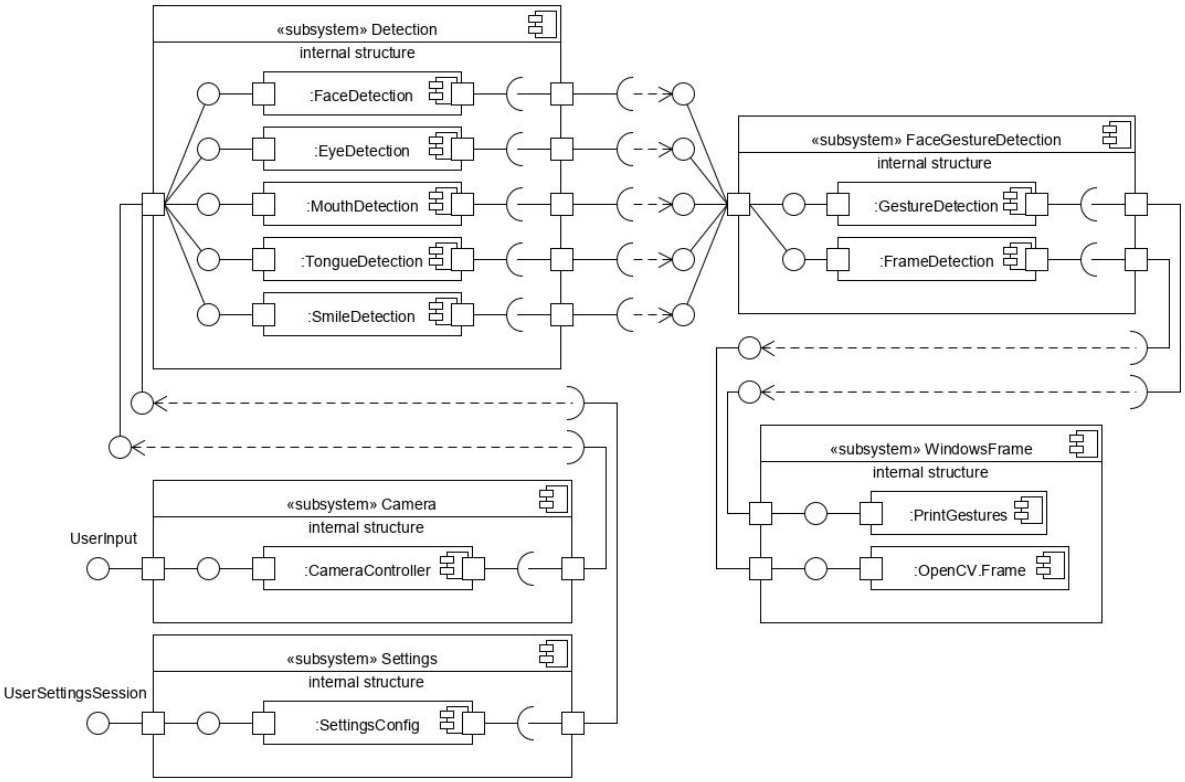
`EyeDetector` - detekuje úkony očí

`detect(Frame)` - upravená funkcia, ktorá zistí, ktorý úkon očí bol vykonaný

`MouthDetector` - detekuje úkony úst

`detect(Frame)` - upravená funkcia, ktorá zistí, ktorý úkon úst bol vykonaný

4.4 UML - Diagram komponentov



5 Špecifikácia vonkajších interfejsov

Aplikácia bude komunikovať a prijímať vstupy z kamery. Môže ísť o internú kameru alebo kameru pripojenú cez USB. Komunikácia bude zabezpečená prostredníctvom knižnice openCV, ktorá pomocou modulu cv2 bude čítať dáta z kamery.

6 Formáty súborov

Systém bude implementovaný v jazyku Python, teda implementačné súbory budú vo formáte .py. Systém bude obsahovať jeden konfiguračný súbor vo formáte .txt na popisovanie gest.

6.1 Konfiguračný súbor

Štruktúra config súboru bude nasledovná: jeden riadok znamená jedno nastavenie. Po spracovaní config súboru si to v programovacom jazyku uložíme do mapy, z ktorej budeme pristupovať k jednotlivým elementom z nastavení. Gestá budú uložené v triede Gesture, kde bude aj zoznam obsahujúci celistvé postupnosti gest.

Idea mapy výpisov je nasledovná (gests predstavuje mapu) : gests[“LRlrmsc”] = “Vypis popis gesta”

- < - pohľad doľava
- > - pohľad doprava
- l - zatvorené ľavé oko krátko
- r - zatvorené pravé oko krátko
- L - zatvorené ľavé oko dlho
- R- zatvorené pravé oko dlho
- M - otvorené ústa
- S - úsmev

Riadky súboru budú definované nasledovne:

- Prvý riadok bude číslo kamery
- Druhý riadok bude určovať dĺžku pauzy v ms, po ktorej sa vyhodnotí gesto
- Tretí riadok bude určovať dĺžku krátkeho úkonu v ms
- Štvrtý riadok bude určovať dĺžku dlhého v ms
- Piaty riadok bude určovať dĺžku zobrazenia výpisu po rozpoznaní gesta
- Každý ďalší riadok bude definovať gesto
    - najprv bude názov gesta v úvodzovkách
    - po medzere výpis po rozpoznaní v úvodzovkách
    - po ďalšej medzere bude postupnosť úkonov označených znakom

  “názov gesta” “výpis” l<MR

7 rozdelenie práce

- UI - Martina
- pomer strán, čierne bary - Martina
- konfiguračný súbor a jeho čítanie - Ivana
- kalibrácia dĺžky žmurkania - Ivana, Martina
- kalibrácia detekcie - Miroslav
- Zameranie sa na jednu osobu - Matej
- kontrola vykonania gesta - Matej
- kontrola dostatku svetla - Miroslav
- testovanie programu s rôznymi kamerami - všetci